UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/600,284	06/20/2003	Steve Burns	50277-2139	9791

42425 7590 11/23/2010
HICKMAN PALERMO TRUONG & BECKER/ORACLE
2055 GATEWAY PLACE
SUITE 550
SAN JOSE, CA 95110-1083

EXAMINER
TSUI, WILSON W

ART UNIT	PAPER NUMBER
2178	

MAIL DATE	DELIVERY MODE
11/23/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	**Application No.** 10/600,284	**Applicant(s)** BURNS ET AL.
	Examiner WILSON TSUI	**Art Unit** 2178

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *27 September 2010*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-23,47 and 49-72* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-23,47 and 49-72* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1. This non-final action is in response to the RCE filed on: 09/27/10.

2. Claims 1-9, 13, 16-20, 23, 47, 49-57, 61, 64-68, 71 and 72 are amended. Claims 24-46 and 48 are cancelled. Claims 1, 18, 49, and 66 are independent claims. Claims 1-23, 47, and 49-72.

3. The following rejections are withdrawn, in view of new grounds of rejection necessitated by applicant's amendments:

- Claims 1-3, 5, 7, 8, 13, 14, 16-22, 49-51, 53-62, and 64-72 rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al, in further view of Schaeck.
- Claims 4 and 52 rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al, in view of Schaeck and further in view of Hind et al.
- Claims 6, 9, 10, 11, 12, 23, and 47 rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al, in view of Schaeck, and further in view of Polizzi et al.
- Claims 15, and 63 rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al, in view of Schaeck, and further in view of Katariya et al.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-3, 5, 7, 8, 13, 14, 16-22, 49-51, 53-62, and 64-72 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in further view of Schaeck2 (US Application: US 2003/0163513 A1, published: Aug. 28, 2003, filed: Feb. 22, 2002).

With regards to claim 1, Abrams et al teaches:

- *receiving a request to display the page* (whereas, a user requests a page: Fig 6)

In response to receiving a request to display the page, performing the steps of:

- *Determining that the page is associated with a page parameter* (Fig 6: whereas a page is displayed according to user preference data/parameters for collecting data from a particular site/URL (column 4, lines 13-30))

- *Inspecting a mapping to determine that the page parameter is mapped to a portlet parameter of a portlet* (As shown in Fig 2a, and Fig 2b, and explained in column 4, lines 13-30, the page parameters are checked and refined by a user (thus establishing mapping data with respect to a page parameter and a portlet parameter) to determine what summary information to display in each of the portlets (Fig 6, 630).

- *Passing a value associated with the page parameter as a value of the portlet parameter to* a routine responsible for rendering the component, *and the* routine *generating the component based upon the value associated with the portlet* *parameter* (Fig 2a, and Fig 2b: whereas, the portlets use page parameters such as URL data to display page/summary information for the page located at the particular URL, and constraint based parameters to display constraint based page/summary information for the page located at the particular URL) *and inserting the component that was generated by the* routine *into the page* (Fig 6: whereas, the generated data/component is inserted into a page).

- *wherein the steps of the method are performed by one or more computing devices* (column 2, lines 52-65: whereas, the method can be implemented on a display computing device).

However, Abrams et al does not expressly teach
generating and storing a mapping that maps one or more page parameter names to one or more portlet parameter names, wherein the mapping is stored separate from pages associated with the one or more page parameters that correspond to the one or more page parameter names; ... determining that the page is associated with a *particular* page parameter *that has a particular page parameter name from the one or more page parameter names, using the mapping to determine which portlet parameter names are mapped to the particular page parameter name; wherein using the mapping includes retrieving* … the mapping; and *wherein the portlet is executable code that is operable to generate page component;* passing a value associated with the *particular* page parameter *name* to the portlet as a value of the *particular* portlet parameter *that corresponds to the particular portlet parameter name*, the portlet generating a

component based upon the value *that is passed to the portlet as the value of* the *particular* portlet parameter; and inserting the component that was generated by the *portlet* into the page;

However, Schaeck2 teaches *generating and storing a mapping that maps one or more page parameter* data *to one or more portlet parameter* data (paragraph 0022: whereas, a page aggregation component refers to page parameter data such as name/access-level-data, specific to a user's role, and finds one or more portlets that are mapped to the specific parameter data (the plurality of portlets, each generating component-type data)),

wherein the mapping is stored separate from pages associated with the one or more page parameters that correspond to the one or more page parameter data (paragraphs 0020, 0024: whereas, externally stored mapping data 520, and XLINK statements are separate from pages, and are used to determine from a plurality of pages, a particular page that is assigned to a particular role/parameter-data).

... determining that the page is associated with a *particular* page parameter *that has a particular page parameter data type from the one or more page parameter data types* (paragraphs 0068, 0069: whereas, from a plurality of role specific pages, a particular composite page is associated with user profile/role) *using the mapping to determine which portlet parameter are mapped to the particular page parameter* (paragraph 0022 : whereas, using block 520 and xlink data, appropriate portlets (with portlet settings/parameters) are retrieved)*;*

wherein using the mapping includes retrieving … the mapping (paragraphs 0022, 0061: using block 520, xlink data, and portal definitions);

and *wherein the portlet is executable code that is operable to generate page component* (paragraph 0053: whereas portlets are executed, and return component data)*;*

passing a value associated with the *particular* page parameter to the portlet as a value of the *particular* portlet parameter *that corresponds to the particular portlet parameter* (paragraph 0053, 0061: whereas through the use of mapping data, appropriate functions based on roles are passed using parameters),

the portlet generating a component based upon the value *that is passed to the portlet as the value of* the *particular* portlet parameter (paragraph 0061, 0062: whereas a portlet can generate a response, based upon request parameters);

and inserting the component that was generated by the *portlet* into the page (paragraph 0044, paragraphs 0082-0084: whereas an aggregate - composite page is generated, by inserting component data);

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's method for passing values to rendering components for generating/displaying a portal with aggregated data; such that the values are passed to portlets (that are identified through mapping information) when generating one or more page components in an aggregated page, as similarly explained by Schaeck2. The combination would have allowed Abrams et al to have "implemented a technique .. that does not require end users to have programming skills" (Schaeck2, paragraph 0015).

However, the combination of Abrams et al and Schaeck2 does not expressly teach the mapped parameter data can be based on parameter *names.*

Yet, as known in the computer science arts, a variable is a symbolic *name*, and a parameter is a type of variable, that can be used as input in subroutines. Schaeck2 further teaches that parameters can be mapped to other parameters (paragraph 0062: whereas, parameters of one service/subroutine, can be mapped to parameters of another service/subroutine). Thus, since parameters are variables, then the parameters referenced in Schaeck2 are also *named*.

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams and Schaeck2's method for selecting portlets from mapping data to generate a page, such that the mapping, which includes parameter type data, can specifically be referenced in terms of parameter mapping (named variable), as also taught by Schaeck2. The combination would have allowed Schaeck2 to have "provided role based views into services which may comprise aggregations of sub-services that span multiple enterprises" (Schaeck2, paragraph 0017).

With regards to claim 2, which depends on claim 1, Abrams et al and Schaeck2 similarly teaches *wherein: using the mapping further includes determining that the particular page parameter name is mapped to a second portlet parameter name associated with a second component of the page; and in response to receiving the request to display the web page, further performing the step of passing the value associated with the particular page parameter name as the value of the second portlet parameter that is associated with a second portlet that generates the second component*, as similarly explained in the rejection for claim 1, whereas page parameter data can be used for a plurality of portlets, to generate an aggregated page comprising a plurality of components).

With regards to claim *3,* which depends on claim 1, Abrams et al and Schaeck2 similarly teach *wherein: the step of generating and storing the mapping further comprises mapping a plurality of page parameter names, corresponding to page parameters for the web page, to a plurality of portlet parameter names corresponding to portlet parameters associated with the component of web page; the step of inspecting the mapping further comprises the step of determining which page parameter names of* *the plurality of page parameter names are mapped to each of the plurality of portlet parameter names; the step of passing the value further comprises the step of passing, based on the mapping, values associated with the plurality of page parameter names as the values of the portlet parameters, corresponding to the plurality of portlet parameter names of the portlet that generates the component; and the step of the portlet generating the component further comprises the step of the portlet generating the component based upon the values associated with the plurality of portlet parameter names*, as similarly explained in the rejection for claim 1 (whereas multiple parameters are supported using link data and definition data to call/pass values to a portlet for component generation), and is rejected under similar rationale.

With regards to claim 5, which depends on claim 1, Abrams et al and Schaeck2 similarly teaches wherein the step of generating and storing the mapping comprises *the steps of mapping the particular page parameter name to the particular portlet parameter name and mapping a second page parameter name to a second portlet parameter name corresponding to a second portlet parameter of the portlet that generates the component of the web page*, as similarly explained in the rejection for claim 1, URL data is the first page parameter, and constraint based parameters are used as secondary parameters for the component of the page; and thus, rejected under similar rationale.

With regards to claim 7, which depends on claim 1, Abrams et al and Schaeck2 similarly teach *wherein the request to display the page includes a URL and the URL includes the value associated with the particular page parameter name, and wherein the step of passing the value associated with the page parameter name is performed by passing the value contained in the URL as the value of the particular portlet parameter* (whereas, as explained in column 4, lines 13-30, and in the rejection for claim 1, URL data is used as parameter information, to be passed as the value of the portlet parameter).

With regards to claim 8, which depends on claim 1, Abrams et al and Schaeck2 teaches *named* parameter mapping, as similarly explained in the rejection for claim 1, and is rejected under similar rationale. Furthermore, the combination of Abrams et al and Schaeck2 teaches *further comprising the steps of: presenting to a user, a user interface for customizing the page; in response to the user interacting with the user interface, obtaining a user specified value for the particular page parameter name; and wherein the step of passing the value associated with the particular page parameter name is performed by passing the user specified value as the value of the particular portlet parameter of the portlet that generates the component*, (Abrams et al, column 4, lines 1-12, and column 6, lines 25-32: a user interface is used by a user to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying the component).

Additionally, as explained in the rejection for claim 1, the combination of Abrams et al and Schaeck2's routine is modified such that a portlet-method is used when *passing the value of the portlet parameter name to the portlet that generates the component*.

With regards to claim 13, which depends on claim 1, Abrams et al and Schaeck2 teaches *named* parameter mapping, as similarly explained in the rejection for claim 1, and is rejected under similar rationale. Furthermore, the combination of Abrams et al and Schaeck2 teaches*further comprising the step of presenting to a page designer a user interface for specifying the mapping between the particular page parameter name and the portlet parameter name* (Abrams et al: whereas, as explained in column 4, lines 1-12, and column 6, lines 25-32, a user interface is used by a user to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying/mapping the component)).

With regards to claim 14, which depends on claim 1, Abrams et al teaches registering the portlet with a portal repository, wherein the process of registering the portlet causes data associated with the portlet to be stored in the portal repository (Abrams et al, claim 1: whereas, a data source comprises registered profile data associated with the routine). Also the combination of Abrams et al and Schaeck2 teaches registering the portlet with a portal repository, wherein the process of registering the portlet causes data associated with the portlet to be stored in the portal repository (Schaeck2, paragraph 0060: whereas portlets are registered in a portal registry).

With regards to claim 16, which depends on claim 1, Abrams et al and Schaeck2 teaches *named* parameter mapping, as similarly explained in the rejection for claim 1, and is rejected under similar rationale. Furthermore, the combination of Abrams et al and Schaeck2 teaches *further comprising the step of receiving input from a page designer through a user interface to create the mapping between the portlet parameter name and the particular page parameter name* (Abrams et al: whereas, as explained in column 4, lines 1-12, and column 6, lines 25-32, a user interface is used by a user to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying/mapping the component).

With regards to claim 17, which depends on claim 1, Abrams et al teaches the method further comprises the step of retrieving the stored value; and the step of the porltet generating the component further comprises the step of the portlet generating the component based upon the retrieved value (claim 1 of Abrams et al, Fig. 2A: whereas, the stored value(s)/preferences/constraints are stored in data stores, which are used to generate the components 240, 250, and 260). Additionally, as explained in the rejection for claim 1, the combination of Abrams et al and Schaeck2 teaches Abrams et al's routine is modified such that *a portlet-method is used when passing the value of the portlet parameter to the portlet that generates the component*.

With regards to claim *18.* Abrams et al teaches a method comprising:

In response to a user manipulating a component associated with a *page, a portlet that generates the component generating a particular event* (column 4, lines 20-21: whereas a user manipulates a web address in component 220 of a portlet, causing the portlet in the page to generate a URL selection event)

*logic associated with the page; inspecting a first mapping (*whereas, as explained in column 4, lines 1-12, and column 6, lines 25-32, a user uses a component/portlet (by generating an event as explained above) to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying/mapping the component*), determining, based on the first mapping and the passed data, an action to perform in response to the particular event* (whereas the action to perform is to display all hyperlinks with their associated text for the selected site in pane 260 (column 4, lines 21-24))*; based on the first mapping, determining that an event output parameter associated with the particular event is mapped to a page parameter; and causing the action to be performed* … wherein *causing the action to be performed comprises passing a value associated with the event output parameter as the value of the particular page parameter; wherein the steps of the method are performed by one or more computing devices* (column 4, lines 21-29: whereas, the URL data that represents the event is mapped to panes 240, 250, and 260, and a display action with regards to the URL data is performed).

However, Abrams et al does not expressly teach *generating and storing a first* *mapping that maps one or more events to one or more actions and one or more event output parameter names to one or more page parameter names, wherein the first mapping is stored separate from web pages associated with the one or more page parameters that correspond to one or more page parameter names*; wherein the web pages include a web page; wherein the portlet is executable code that is operable to generate page components; logic associated with the *web* page *intercepting data, passed by the portlet, that represents the particular event; retrieving* the first mapping; determining that an event output parameter *name, which corresponds to an event output parameter* associated with the particular event is mapped to the particular page parameter *name;* passing a value associated with the event output parameter *name* as the value of *a particular page parameter that corresponds to* the particular page parameter *name*.

Yet, Schaeck2 teaches *generating and storing a first mapping that maps one or more events to one or more actions* (paragraph 0062: whereas one or more transition events are mapped to a subsequent operation/action) *and one or more event output parameters names to one or more page parameters* (paragraph 0062: whereas, the event is detected for a role-specific page)*, wherein the first mapping is stored separate from web pages associated with the one or more page parameters that correspond to the one or more page parameters* (paragraph 0062: whereas, a separate data mapping is implemented for linking operations); wherein the web pages include a web page (paragraphs 0006, 0007: whereas, web pages are generated and aggregated through portal pages and/or web services); wherein the portlet is executable code that is operable to generate page components (paragraph 0053: whereas, the portlet is executed and returns component data); logic associated with the *web* page *intercepting data, passed by the portlet, that represents the particular event* (paragraph 0062: whereas, transition events/conditions are intercepted/detected for the web page)*; retrieving* the first mapping (paragraph 0062: whereas, data mapping is retrieved to perform the appropriate data operations/actions); determining that an event output parameter *name, which corresponds to an event output parameter* associated with the particular event is mapped to the particular page parameter*;* passing a value associated with the event output parameter *name* as the value of *a particular page parameter that corresponds to* the particular page parameter *name* (paragraph 0062: whereas, based on the data mapping, values are passed to the appropriate service/page-service).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's method for passing values to rendering components for generating/displaying a portal with aggregated data; such that the values are passed to portlets (that are identified through mapping information) when generating one or more page components in an aggregated page, as similarly explained by Schaeck2. The combination would have allowed Abrams et al to have "implemented a technique .. that does not require end users to have programming skills" (Schaeck2, paragraph 0015).

However, the combination of Abrams et al and Schaeck et al do not expressly teach the mapped parameter data can be based on parameter *names*.

Yet, as known in the computer science arts, a variable is a symbolic *name*, and a parameter is a type of variable, that can be used as input in subroutines. Schaeck2 further teaches that parameters can be mapped to other parameters (paragraph 0062: whereas, parameters of one service/subroutine, can be mapped to parameters of another service/subroutine). Thus, since parameters are variables, then the parameters referenced in Schaeck2 are also *named*.

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams and Schaeck2's method for selecting portlets from mapping data to generate a page, such that the mapping, which includes parameter type data, can specifically be referenced in terms of parameter mapping (named variable), as also taught by Schaeck2. The combination would have allowed Schaeck2 to have "provided role based views into services which may comprise aggregations of sub-services that span multiple enterprises" (Schaeck2, paragraph 0017).

With regards to claim 19, which depends on claim 18, Abrams et al teaches *wherein the page is a first page and the particular page parameter is associated with a second page; and the step of causing the action to be performed further comprises the step of passing the value of the particular page parameter to logic responsible for rendering a second page*, as similarly explained in claim 1, the URL data (value of the page parameter) is passed to logic/routine(s) responsible for rendering an updated/second page, and is rejected under the similar rationale.

With regards to claim 20, which depends on claim 18, Abrams et al teaches *wherein the step of causing the action to be performed further comprises the step of generating a request that specifies a URL, wherein the value of the particular page* *parameter is included in the URL*: (whereas, as explained in column 4, lines 13-30, and in the rejection for claim 1, URL data is used as parameter information, to be passed as the value of the portlet parameter).

With regards to claim 21, which depends on claim 20, Abrams et al teaches*: the step of generating the request further comprises the step of generating a request for executable code; and the step of causing the action to be performed further comprises the step of invoking the executable code*, as similarly explained in the rejection for claim 1, page parameter data is passed to the appropriate portlet parameters, and the passing of the value causes the display/render action to be performed. Since the rendering as shown in Fig 2A as performed/executed, the figure inherently teaches that code must have been executed in order for the appropriate components/portlets to have been updated with the mapped parameter values.

With regards to claim 22, which depends on claim 21, Abrams et al teaches *wherein the executable code is a web service* (column 1, lines 45-60: whereas, the executable code, provides user's with a service to collect information from disparate sources, to be displayed in a summarized and consistent manner).

With regards to claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 1, is rejected under similar rationale.

With regards to claim 50, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more

processors perform a method similar to the method of claim 2, is rejected under similar rationale.

With regards to claim 51, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 3, is rejected under similar rationale.

With regards to claim 53, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 5, is rejected under similar rationale.

With regards to claim 54, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 6, is rejected under similar rationale.

With regards to claim 55, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 7, is rejected under similar rationale.

With regards to claim 56, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 8, is rejected under similar rationale.

With regards to claim 57, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 9, is rejected under similar rationale.

With regards to claim 58, which depends on claim 57, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 10, is rejected under similar rationale.

With regards to claim 59, which depends on claim 57, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 11, is rejected under similar rationale.

With regards to claim 60, which depends on claim 57, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 12, is rejected under similar rationale.

With regards to claim 61, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 13, is rejected under similar rationale.

With regards to claim 62, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more

processors perform a method similar to the method of claim 14, is rejected under similar rationale.

With regards to claim 64, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 16, is rejected under similar rationale.

With regards to claim 65, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 17, is rejected under similar rationale.

With regards to claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 18, is rejected under similar rationale.

With regards to claim 67, which depends on claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 19, is rejected under similar rationale.

With regards to claim 68, which depends on claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 20, is rejected under similar rationale.

With regards to claim 69, which depends on claim 68, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 21, is rejected under similar rationale.

With regards to claim 70, which depends on claim 69, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 22, is rejected under similar rationale.

With regards to claim 71, which depends on claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 23, is rejected under similar rationale.

With regards to claim 72, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 47, is rejected under similar rationale.

5. Claims 4 and 52 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in view of Schaeck2 (US Application: US 2003/0163513 A1, published: Aug. 28, 2003, filed: Feb. 22, 2002), and further in view of Hind et al (US Application: 2004/0205555 A1, published: Oct. 14, 2004, filed: Sep. 18, 2001).

With regards to claim 4, which depends on claim 1, Abrams et al and Schaeck2 teaches wherein the step of generating and storing the mapping comprises the step of *mapping the particular page parameter name to the particular portlet parameter name associated with the component of the web page*, as similarly explained in the rejection for claim 1, and is rejected under similar rationale. However, Abrams et al does not expressly teach … *without mapping the particular page parameter name to portlet parameter names associated with any other components of the page*.

Hind et al teaches … *without mapping the particular parameters to portlet parameters associated with any other components of the page* (Fig. 3A, paragraph 0024: whereas, content for some components/portlets are updated, while some are not, and thus components/portals are selectively mapped for receiving parameter data).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al and Hofmann et al's method for mapping page parameter names, to have only mapped page parameter names to a select page component, as taught by Hind et al. The combination of Abrams et al, Schaeck2, and Hind et al would have allowed Abrams et al to have "reduced the time a user waits for receiving a portal page [by] spawning individual threads for reach portlet" (Hind et al, paragraph 0009).

With regards to claim 52, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more

processors perform a method similar to the method of claim 4, is rejected under similar rationale.

6. Claims 6, 9, 10, 11, 12, 23, and 47 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in view of Schaeck2 (US Application: US 2003/0163513 A1, published: Aug. 28, 2003, filed: Feb. 22, 2002), and further in view of Polizzi et al (US Application: US 2002/0052954 A1, published: May 2, 2002, filed: Apr. 27, 2001)

With regards to claim 6, Abrams et al teaches *establishing the particular page parameter name, and passing the value associated with the particular page parameter name further comprises the step of passing the value as the value of the particular portlet parameter of the portlet that generates the component,* as similarly explained in the rejection for claim 1, and is rejected under similar rationale. However, Abrams et al does not expressly teach the value associated with the page parameter is a *default value*.

Yet, Polizzi et al teaches a *default value* (paragraph 0033: whereas, default values can be associated with page parameters, when retrieving the appropriate portlet).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al and Schaeck2's method for mapping a page parameter name to a portlet parameter name, such that the page parameter value is a default value, as also taught by Schaeck2. The combination of Abrams et al, Schaeck2, Polizzi et al would have allowed Abrams et al to have “allowed a user to have been able to customize the content of his personal portal page by adding or removing certain portal objects or by modifying the content of certain portal objects” (Polizzi et al, paragraph 0030).

With regards to claim 9, which depends on claim 1, Abrams et al teaches *determining a selected value based on override* preferences (column 6, lines 12-32: whereas, override settings/preferences are determined), and *passing the selected value as the value of the particular portlet parameter of the routine responsible for rendering the component* (as similarly explained in the claim 1, and also in column 6, lines 12-32, the selected preference values are used to render a customized view).

However Polizzi et al teaches a override *hierarchy,* and passing the user specified value as the value of the portlet parameter to the portlet that generates the component (paragraph 0033: whereas, a user specified value can override a default value of the portlet parameter to the portlet that generates the component).

It would have been obvious to one of the one of the ordinary skill in the art at the time of the invention to have modified. Abrams et al method for using portlets to assemble a page, such that one or more portlets can accept user specified parameters that override previous parameter values, as taught by Polizzi et al. The combination of Abrams et al, Schaeck2, and Polizzi et al would have allowed Abrams et al to have have "implemented a standardized, easy-to-learn method for retrieving data through the use of an enterprise-wide computer system, which is connected to a variety of computer systems” (Polizzi et al, paragraph 0003).

With regards to claim 10, which depends on claim 9, Abrams et al teaches *the plurality of values includes a URL page parameter value* (as similarly explained in the rejection for claim 1) *and a customize page parameter value* (as similarly explained in the rejection for claim 1, whereas the constraint based parameters, are custom page parameter values), as well as *override* preferences (column 6, lines 12-32: whereas, override settings/preferences are determined). However, Abrams et al does not expressly teach an override *hierarchy that specifies that the URL page is the page parameter value is the selected value.*

Yet, the combination of Abrams et al, Schaeck2, and Polizzi et al teaches an override hierarchy, and a *default page parameter value as the selected value,* as similarly explained in the rejection for claim 9, and is rejected under similar rationale.

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's page parameter value, such as a URL; such that the override hierarchy is used to specify the value to be a selected value, as taught by Polizzi et al. The combination of Abrams et al, Schaeck2, and Polizzi et al would have allowed Abrams et al to have “allowed a user to customize content of his personal page, by adding or removing certain portal objects, or by modifying the content of certain portal objects” (Polizzi et al, paragraph 0030).

With regards to claim 11, which depends on claim 9, Abrams et al teaches *the plurality of values includes a page parameter value* (as similarly explained in the rejection for claim 1) *and a customize page parameter value* (as similarly explained in the rejection for claim 1, whereas the constraint based parameters, are custom page parameter values), as well as *override* preferences (column 6, lines 12-32: whereas, override settings/preferences are determined). However, Abrams et al does not expressly teach an override *hierarchy that specifies that the customize page parameter value is the page parameter value is the selected value.*

Yet, the combination of Abrams et al, Schaeck2, and Polizzi et al teaches an override hierarchy that specifies that the customized page parameter value is the page parameter value is the selected value, as similarly explained in the rejection for claim 9, and is rejected under similar rationale.

With regards to claim 12, which depends on claim 9, Abrams et al teaches *the plurality of values includes a page parameter value* (as similarly explained in the rejection for claim 1),a *portlet specified value* (as similarly explained in the rejection for claim 1), as well as *override* preferences (column 6, lines 12-32: whereas, override settings/preferences are determined). However, Abrams et al does not expressly teach, the page parameter value is a *default value, and an* override *hierarchy that specifies that the default page parameter is the selected value.*

Yet, the combination of Abrams et al, Schaeck2, and Polizzi et al teaches *an override hierarchy that specifies that the default page parameter is the selected value,* as similarly explained in the rejection for claim 9, and is rejected under similar rationale.

With regards to claim *23*, which depends on claim 18, Abrams et al and Schaeck2 teaches wherein*:*

The action comprises rendering a second page, wherein the particular page parameter is associated with the second page, and wherein rendering the second page (as similarly explained in the rejection for claim 19, and is rejected under similar rationale) comprises the steps of:

Inspecting a mapping to determine that the particular page parameter name is mapped to a particular portlet parameter name that corresponds to a particular portlet, as similarly explained in the rejection for claim 1, and is rejected under similar rationale.

Passing the value of the particular page parameter as the value fo the particular portlet parameter of a portlet, that corresponds to the particular portlet parameter name, as similarly explained in the rejection for claim 1, and is rejected under similar rationale.

The second portlet generating a component based upon the value associated with the particular portlet parameter, as similarly explained in the rejection for claim 19, and is rejected under similar rationale.

Inserting the second component that was generated by a portlet into a page, as similarly explained in the rejection for claim 19, and is rejected under similar rationale.

However, Abrams et al does not expressly teach Inspecting a *second mapping* to determine that the page parameter name is mapped to a portlet parameter name of a *second* portlet that generates a *second* component of the *second* page that is based, at least in part, on the particular portlet parameter; and Passing the value of the particular portlet parameter as the value of the particular portlet parameter, of the *second* portlet.

Yet, the Polizzi et al similarly teaches Inspecting a *second mapping* to determine that the page parameter is mapped to a portlet parameter of a *second* portlet that generates a *second* component of the *second* page that is based, at least in part, on the particular portlet parameter; and Passing the value of the page parameter as the value of the particular portlet parameter to the *second* portlet (paragraph 0092 of Polizzi et al: whereas, more than one portlet is retrieved to generate a page comprising a plurality of components);

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al and Schaeck2's Portal aggregation method of using a separately stored mapping to map a parameter name to generate a first component, such that the mapping would further include mapping the parameter name to generate a second component, as taught by Polizzi et al. The combination would have "allowed a user to customized content of his personal page, by adding or removing certain portal objects, or by modifying the content of certain portal objects" (Polizzi et al, paragraph 0030).

With regards to claim 47, which depends on claim 1, the combination of Abrams et al, Schaeck2, and Polizzi et al teaches *wherein the portlet is a first portlet and wherein the mapping maps a single page parameter name, of the one or more page parameter names, to a first portlet parameter name corresponding to a first portlet parameter of the first portlet, and to a second portlet parameter name corresponding to a second portlet,* as similarly explained in the rejection for claim 23, and is rejected under similar rationale.

7. Claims 15, and 63 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in view of Schaeck2 (US Application: US 2003/0163513 A1, published: Aug. 28, 2003, filed: Feb. 22, 2002) , and further in view of Katariya et al (US Patent: 6,564,251 B2, issued: May 13, 2003, filed: Dec. 3, 1998).

With regards to claim 15, which depends on claim 14, Abrams et al, and Schaeck2 teaches *the data associated with the portlet,* and *communicated with the portal repository*, as similarly explained in the rejection for claim 14, and is rejected under similar rationale. However, Abrams et al, and Schaeck2 do not expressly teach the data associated with the portlet, is communicated to the portal repository *as an XML document*.

Katariya et al teaches communicating with the portal repository, through the use of an *XML document* (columns 5 and 6, lines 59-67 and 1-9 respectively: whereas, preference/parameter information is communicated to a portal repository via XML format).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's communication of data to a portal repository, such that the data is passed using a XML document, as similarly taught by Katariya et al. The combination of Abrams et al, Schaeck2, and Katariya et al would have allowed Abrams et al to have allowed "the content of each page to have been enhanced by the rendered data from the provider objects, thereby adding dynamic behavior to the predefined page" (Katariya et al, column 2, lines 26-31).

With regards to claim 38, for a computer-readable storage medium storing one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform a method similar to the method recited in claim 15, and is rejected under similar rationale.

With regards to claim 63, which depends on claim 62, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 15, is rejected under similar rationale.

Response to Arguments

8. Applicant's arguments with respect to claims 1-23, 47, and 49-72 have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to WILSON TSUI whose telephone number is (571)272-7596. The examiner can normally be reached on Monday - Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Stephen Hong can be reached on (571) 272-4124. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Wilson Tsui/
Patent Examiner
Art Unit: 2178
November 19, 2010

	/CESAR B PAULA/ Primary Examiner, Art Unit 2178